功能：反算三次B样条插值开曲线控制顶点。
输入参数：(m_xDVertex,m_yDVertex[i])-为受保护成员,i=0,1,...,n-2存数据点,i=n-1存首端显示切矢矢端,i=n存末端显示切矢矢端。m_eType0,m_eType1-首末边界条件标识码。
输出参数：(m_xAVertex,m_yAVertex)-控制顶点，为受保护成员。
调用函数：InitEdge-置首末边界条件系数与右端矢量；delta-节点值向前差分。

```
void BTridia()
{
	int n= m_xDVertex.GetSize()-1;
	int k=2;
	m_xAVertex.SetSize(n+1), m_yAVertex.SetSize(n+1);	//i=0,1,...,n存控制顶点
	m_xAVertex[0]=m_xDVertex[0];
	m_yAVertex[0]=m_yDVertex[0];
	m_xAVertex[n]=m_xDVertex[n-2];
	m_yAVertex[n]=m_yDVertex[n-2];
	double a1,b1,c1,e1x,e1y,an_1,bn_1,cn_1,en_1x,en_1y;
	InitEdge(b1,c1,a1,e1x,e1y,cn_1,an_1,bn_1,en_1x,en_1y);
CArray<CArray<double,double>,CArray<double,double>&> dia;//系数矩阵存三对角元素
	dia.SetSize(3);
	//系数矩阵含第1，2,...,n-1行，行数为n-1。第i 行dia[0][i],dia[1][i],dia[2][i]存ai,bi,ci
	//三个元素。因系数矩阵不含第0行，dia[0][0],dia[1][0],dia[2][0]三元素空置。
	//系数矩阵既有三对角元素，又首行多出a1与末行多出cn_1两个元素。
	for(int i=0;i<3;i++) dia[i].SetSize(n);
	CArray<CArray<double,double>,CArray<double,double>&> t;
	t.SetSize(k);
	for(i=0;i<k;i++) t[i].SetSize(n);
	//生成系数矩阵第2，3，...,n-2行元素与右端矢量(t[0][i], t[1][i]),i=2，3，...,n-2
	for(i=2;i<=n-2;i++)    //函数double delta(int i);在头文件中声明为受保护(protected)
	{                      //成员，用于计算节点矢量中节点值的一阶向前差分
		dia[0][i]=(delta(i+2)*delta(i+2))/(delta(i)+delta(i+1)+delta(i+2));
		dia[1][i]=(delta(i+2)*(delta(i)+delta(i+1)))/(delta(i)+delta(i+1)+delta(i+2))
  +(delta(i+1)*(delta(i+2)+delta(i+3)))/(delta(i+1)+delta(i+2)+delta(i+3));
		dia[2][i]=(delta(i+1)*delta(i+1))/(delta(i+1)+delta(i+2)+delta(i+3));
		t[0][i]=(delta(i+1)+delta(i+2))*m_xDVertex[i-1];
		t[1][i]=(delta(i+1)+delta(i+2))*m_yDVertex[i-1];
	}
	dia[0][1]=a1; dia[1][1]=b1;    dia[2][1]=c1;            //首行边界条件系数矩阵三元素
	dia[0][n-1]=an_1; dia[1][n-1]= bn_1;dia[2][n-1]=cn_1;  //末行边界条件系数矩阵三元素
	t[0][1]=e1x;  t[1][1]=e1y;                          //首行边界条件右端首点矢量。
	t[0][n-1]=en_1x; t[1][n-1]=en_1y;                   //末行边界条件右端末点矢量。
	if(m_eType0!=5&&m_eType1!=5)
	{
		if(n>3)
		{
			dia[0][1]=-dia[0][1]/dia[2][2];             //开始消元过程，将首行a1消为0
			dia[1][1]=dia[1][1]+dia[0][2]*dia[0][1];
			dia[2][1]=dia[2][1]+dia[1][2]*dia[0][1];
			t[0][1]=t[0][1]+t[0][2]*dia[0][1];
			t[1][1]=t[1][1]+t[1][2]*dia[0][1];
			dia[0][1]=0.;
			dia[2][n-1]=-dia[2][n-1]/dia[0][n-2];       //将末行cn_1消为0
			dia[0][n-1]=dia[0][n-1]+dia[1][n-2]*dia[2][n-1];
			dia[1][n-1]=dia[1][n-1]+dia[2][n-2]*dia[2][n-1];
			t[0][n-1]=t[0][n-1]+t[0][n-2]*dia[2][n-1];
			t[1][n-1]=t[1][n-1]+t[1][n-2]*dia[2][n-1];
			dia[2][n-1]=0.;                                                    //至此，系数矩阵
已成为标准三对角矩阵
		}
		for(i=2;i<=n-1;i++)                         //将下三角元素全部消为0
		{
			dia[0][i]=-dia[0][i]/dia[1][i-1];
			dia[1][i]=dia[1][i]+dia[2][i-1]*dia[0][i];
			t[0][i]=t[0][i]+t[0][i-1]*dia[0][i];
			t[1][i]=t[1][i]+t[1][i-1]*dia[0][i];
		}
		for(int l=0;l<k;l++) t[l][n-1]=t[l][n-1]/dia[1][n-1];   //得第n-1个控制顶点
		for(i=n-2;i>=1;i--)                          //回代求得第n-2,n-3,...,1个控制顶点
		{
			for(l=0;l<k;l++) t[l][i]=(t[l][i]-dia[2][i]*t[l][i+1])/dia[1][i];
		}
	}
	//当用于非节点条件，消元时可能出现主对角元素为0，作相应处理。
	if(m_eType0==5||m_eType1==5)
	{
		dia[0][n-2]=dia[0][n-2]-dia[2][n-1]*dia[2][n-2]/dia[1][n-1];  //将dia[2][n-2]消为0;
```

```
                dia[1][n-2]=dia[1][n-2]-dia[0][n-1]*dia[2][n-2]/dia[1][n-1];
                t[0][n-2]=t[0][n-2]-t[0][n-1]*dia[2][n-2]/dia[1][n-1];
                t[1][n-2]=t[1][n-2]-t[1][n-1]*dia[2][n-2]/dia[1][n-1];
                dia[2][n-2]=0.;
                dia[1][2]=dia[1][2]-dia[2][1]*dia[0][2]/dia[1][1];          //dia[0][2]消为0;
                dia[2][2]=dia[2][2]-dia[0][1]*dia[0][2]/dia[1][1];
                t[0][2]=t[0][2]-t[0][1]*dia[0][2]/dia[1][1];
                t[1][2]=t[1][2]-t[1][1]*dia[0][2]/dia[1][1];
                dia[0][2]=0.;
                for(i=3;i<=n-2;i++)                                         //将下三角元素消为0
                {
                        dia[0][i]=-dia[0][i]/dia[1][i-1];
                        dia[1][i]=dia[1][i]+dia[2][i-1]*dia[0][i];
                        t[0][i]=t[0][i]+t[0][i-1]*dia[0][i];
                        t[1][i]=t[1][i]+t[1][i-1]*dia[0][i];
                        dia[0][i]=0.;
                }
                t[0][n-2]=t[0][n-2]/dia[1][n-2];                //得顶点(m_xAVertex[n-2],
m_yAVertex[n-2])
                t[1][n-2]=t[1][n-2]/dia[1][n-2];
                for(i=n-3;i>=2;i--)
                {   //回代求顶点(m_xAVertex[n-3], m_yAVertex[n-3]),...,(m_xAVertex[2], m_yAVertex[2])
                        t[0][i]=(t[0][i]-dia[2][i]*t[0][i+1])/dia[1][i];
                        t[1][i]=(t[1][i]-dia[2][i]*t[1][i+1])/dia[1][i];
                }
                //回代求顶点(m_xAVertex[1], m_yAVertex[1])
                t[0][1]=(t[0][1]-dia[2][1]*t[0][2]-dia[0][1]*t[0][3])/dia[1][1];
t[1][1]=(t[1][1]-dia[2][1]*t[1][2]-dia[0][1]*t[1][3])/dia[1][1];
                //求顶点(m_xAVertex[n-1], m_yAVertex[n-1]
                t[0][n-1]=(t[0][n-1]-dia[2][n-1]*t[0][n-3]-dia[0][n-1]*t[0][n-2])/dia[1][n-1];
                t[1][n-1]=(t[1][n-1]-dia[2][n-1]*t[1][n-3]-dia[0][n-1]*t[1][n-2])/dia[1][n-1];
        }
        //得第1，2，...,n-1个控制顶点
        for(i=1;i<=n-1;i++)
        {
                m_xAVertex[i]=t[0][i];
                m_yAVertex[i]=t[1][i];
        }
        if(m_eType0!=2&&m_eType0!=3)
        {   //该段解决拖动显示首端切矢矢端(m_xDVertex[n-1],m_yDVertex[n-1])
                double px,py;
                int rank=1,nTime=3;
                double u=0.;
                GetDerivat(rank,nTime,u,px,py);
                m_ftx=px;
                m_fty=py;
                m_xDVertex[n-1]=m_xDVertex[0]+0.5*delta(3)*m_ftx;
                m_yDVertex[n-1]=m_yDVertex[0]+0.5*delta(3)*m_fty;
        }
        if(m_eType1!=2&&m_eType1!=3)
        {   //该段解决拖动显示末端切矢矢端(m_xDVertex[n],m_yDVertex[n])
                double px,py;
                int rank=1,nTime=3;
                double u=1.;
                GetDerivat(rank,nTime,u,px,py);
                m_ltx=px;
                m_lty=py;
                m_xDVertex[n]=m_xDVertex[n-2]+0.5*delta(n)*m_ltx;
                m_yDVertex[n]=m_yDVertex[n-2]+0.5*delta(n)*m_lty;
        }
        return;
}
```